

MetaRule

Impersonation - Carefully check for call failure to avoid unintended privilege escalation

Sean Barnum, Cigital, Inc. [vita¹]

Copyright © 2007 Cigital, Inc.

2007-03-29

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 10723 bytes

Attack Category	<ul style="list-style-type: none">• Privilege Exploitation																						
Vulnerability Category	<ul style="list-style-type: none">• Privilege escalation problem• Process management																						
Software Context	<ul style="list-style-type: none">• Security• Process Management																						
Location																							
Description	<p>There are several APIs that are used to temporarily impersonate another security context, often to reduce privilege prior to execution of some capability.</p> <p>If these API calls fail, any further execution of requests could lead to unintended escalation of privilege. It is important to check for failure and not simply continue execution of requests with the assumption that the security context has been changed.</p> <p>If the calling process is running as a highly privileged account, such as LocalSystem, or as a member of an administrative group, the user may be able to perform actions that would otherwise be disallowed.</p>																						
APIs	<table border="1"><thead><tr><th>Function Name</th><th>Comments</th></tr></thead><tbody><tr><td>CAccessToken::Impersonate</td><td>CAccessToken (ATL)</td></tr><tr><td>CAccessToken::ImpersonateLoggedOnUser</td><td>CAccessToken (ATL)</td></tr><tr><td>CoImpersonateClient</td><td>objbase.h</td></tr><tr><td>ImpersonateDdeClientWindow</td><td>ddeml.h</td></tr><tr><td>ImpersonateLoggedOnUser</td><td>winbase.h</td></tr><tr><td>ImpersonateNamedPipeClient</td><td>winbase.h</td></tr><tr><td>ImpersonateSecurityContext</td><td>spi.h</td></tr><tr><td>ImpersonateSelf</td><td>winbase.h</td></tr><tr><td>DdeImpersonateClient</td><td>ddeml.h</td></tr><tr><td>RpcImpersonateClient</td><td>rpc.h</td></tr></tbody></table>	Function Name	Comments	CAccessToken::Impersonate	CAccessToken (ATL)	CAccessToken::ImpersonateLoggedOnUser	CAccessToken (ATL)	CoImpersonateClient	objbase.h	ImpersonateDdeClientWindow	ddeml.h	ImpersonateLoggedOnUser	winbase.h	ImpersonateNamedPipeClient	winbase.h	ImpersonateSecurityContext	spi.h	ImpersonateSelf	winbase.h	DdeImpersonateClient	ddeml.h	RpcImpersonateClient	rpc.h
Function Name	Comments																						
CAccessToken::Impersonate	CAccessToken (ATL)																						
CAccessToken::ImpersonateLoggedOnUser	CAccessToken (ATL)																						
CoImpersonateClient	objbase.h																						
ImpersonateDdeClientWindow	ddeml.h																						
ImpersonateLoggedOnUser	winbase.h																						
ImpersonateNamedPipeClient	winbase.h																						
ImpersonateSecurityContext	spi.h																						
ImpersonateSelf	winbase.h																						
DdeImpersonateClient	ddeml.h																						
RpcImpersonateClient	rpc.h																						

1. <http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html> (Barnum, Sean)

Method of Attack	Elevation of privilege. These calls are frequently used to reduce privilege prior to execution of some capability. However, if any of these calls fail, then the program continues to run at the elevated privilege and is thereby subject to privilege escalation problems.					
Exception Criteria	None noted.					
Solutions	<table border="1"> <thead> <tr> <th data-bbox="807 371 1110 416">Solution Applicability</th> <th data-bbox="1123 371 1426 416">Solution Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="807 425 1110 676">Generally applicable.</td> <td data-bbox="1123 425 1426 676">Always check the return value of the call, and if it fails do not continue execution of the client request as the security context will remain that of the current process.</td> </tr> </tbody> </table>	Solution Applicability	Solution Description	Generally applicable.	Always check the return value of the call, and if it fails do not continue execution of the client request as the security context will remain that of the current process.	
Solution Applicability	Solution Description					
Generally applicable.	Always check the return value of the call, and if it fails do not continue execution of the client request as the security context will remain that of the current process.					
Signature Details	<p>Standard calls to any of the enumerated APIs, such as:</p> <p>BOOL ImpersonateSelf(SEcurity_IMPERSONATION_LEVEL ImpersonationLevel); BOOL DdeImpersonateClient(HCONV hConv); RPC_STATUS RPC_ENTRY RpcImpersonateClient(RPC_BINDING_HANDLE BindingHandle);</p>					
Examples of Incorrect Code	<pre>// No check on ret { WDML_CONV* pConv; HCONV hConv; BOOL ret = FALSE; TRACE("(%p)\n", hConv); EnterCriticalSection(&WDML_CritSect); pConv = WDML_GetConv(hConv, TRUE); if (pConv) { ret = DdeImpersonateClient(hConv); } LeaveCriticalSection(&WDML_CritSect); return ret; }</pre> <pre>/* No check on call */ /* Obtain the security descriptor. */ if (!GetFileSecurity (path, OWNER_SECURITY_INFORMATION GROUP_SECURITY_INFORMATION DACL_SECURITY_INFORMATION,</pre>					

```
pSD, nLength, &nLength))
{
printf ("Unable to obtain
security descriptor.");
return (0);
}
/* Perform security impersonation
of the user and open */
/* the resulting thread token. */
return_bool = ImpersonateSelf
(SecurityImpersonation);
[...]
```

```
// Return code not checked
DWORD dwGuiThreadId = 0;

int UserMessageBox(
RPC_BINDING_HANDLE h,
LPSTR lpszWindowStation,
LPSTR lpszDesktop,
LPSTR lpszText,
LPSTR lpszTitle,
UINT fuStyle)
{
DWORD dwThreadId;
HWINSTA hwinstaSave;
HDESK hdeskSave;
HWINSTA hwinstaUser;
HDESK hdeskUser;
int result;

// Ensure connection to service
window station and desktop, and
// save their handles.

GetDesktopWindow();
hwinstaSave =
GetProcessWindowStation();
dwThreadId =
GetCurrentThreadId();
hdeskSave =
GetThreadDesktop(dwThreadId);

// Impersonate the client and
connect to the User's
// window station and desktop.
RpcImpersonateClient(h);
hwinstaUser =
OpenWindowStation(lpszWindowSt-
ation, FALSE, MAXIMUM_ALLOWED);
if (hwinstaUser == NULL)
{
RpcRevertToSelf();
return 0;
}
```

	<pre>SetProcessWindowStation(hwinst- aUser); [...]</pre>
<p>Examples of Corrected Code</p>	<pre>{ WDML_CONV* pConv; HCONV hConv; BOOL ret = FALSE; TRACE("(%p)\n", hConv); EnterCriticalSection(&WDML_CritSect); pConv = WDML_GetConv(hConv, TRUE); if (pConv) { ret = DdeImpersonateClient(hConv); /* Check ret value here and perform algorithm appropriate response */ if (ret != 0) {RevertToSelf(); /*handle error*/ } } LeaveCriticalSection(&WDML_CritSect); return ret; } [...]</pre> <pre>/* Obtain the security descriptor. */ if (!GetFileSecurity (path, OWNER_SECURITY_INFORMATION GROUP_SECURITY_INFORMATION DACL_SECURITY_INFORMATION, pSD, nLength, &nLength)) { printf ("Unable to obtain security descriptor."); return (0); } /* Perform security impersonation of the user and open */ /* the resulting thread token. */ if (!ImpersonateSelf (SecurityImpersonation)) { printf ("Unable to perform security impersonation."); return (0); } [...]</pre>

	<pre>// Checks return code status = RpcImpersonateClient(Binding); if (status != RPC_S_OK) { #ifdef _DEBUG DisplayError("RpcImpersonateClient", TRUE); #endif return(RPC_S_ACCESS_DENIED); } </pre>				
Source References	<ul style="list-style-type: none"> Howard, Michael & LeBlanc, David C. <i>Writing Secure Code, 2nd ed.</i> Redmond, WA: Microsoft Press, 2002, ISBN: 0735617228. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsecure/html/appsec.asp² http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secauthz/security/impersonateself.asp³ http://lists.samba.org/archive/samba-technical/2003-August/031368.html http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/sec_winui.asp⁵ http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/dataexchange/dynamicdataexchangemanagementlibrary/dynamicdataexchangemanagementreference/dynamicdataexchangemanagementfunctions/ddeimpersonateclient.asp⁶ http://msdn.microsoft.com/library/default.asp?url=/library/en-us/rpc/rpc/rpcimpersonateclient.asp⁷ http://groups-beta.google.com/group/microsoft.public.win32.programmer.networks/browse_thread/thread/52a5e9311cff8595/93fd5b5e62828b0d?q=rpcimpersonateclient+check+return&rnum=2&hl=en#93fd5b5e62828b0d⁸ 				
Recommended Resource					
Discriminant Set	<table border="1"> <tr> <td data-bbox="798 1653 1117 1713"> Operating System </td> <td data-bbox="1117 1653 1442 1713"> <ul style="list-style-type: none"> Windows </td> </tr> <tr> <td data-bbox="798 1713 1117 1812"> Languages </td> <td data-bbox="1117 1713 1442 1812"> <ul style="list-style-type: none"> C C++ </td> </tr> </table>	Operating System	<ul style="list-style-type: none"> Windows 	Languages	<ul style="list-style-type: none"> C C++
Operating System	<ul style="list-style-type: none"> Windows 				
Languages	<ul style="list-style-type: none"> C C++ 				

Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at copyright@cigital.com¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@cigital.com>